A Supervised Learning Approach to Predicting Multigrid Convergence

Nicolas Nytko

Matthew West, Luke Olson, Scott MacLachlan

March 28, 2021

Introduction

- AMG methods are among fastest today for solving sparse linear systems
- Optimal setup for AMG can be difficult, and incorrect parameters could prevent convergence.
  - Put careful thought into the problem and selecting relaxation weights, AMG parameters
  - ~~Just try different values and see what works~~
- For a specific AMG setup, can we predict efficacy ahead of time?
- Look at predicting rate of convergence for specific Poisson, Convection-Diffusion problems.

Poisson Problem

- Look at the 1D variable coefficients case w/ homogeneous Dirichlet conditions

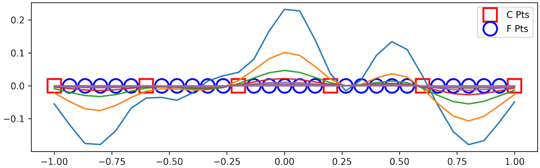
$$-\nabla \cdot (k(\mathbf{x}) \nabla \mathbf{u}) = f$$

$$\Omega = [-1, 1] \quad \mathbf{u}(\partial\Omega) = 0$$

- Discretized on $N = 31$ internal points using finite differences, $k(\mathbf{x})$ is discretized on midpoints to preserve symmetry.
- For arbitrary C/F splitting, can we predict convergence rate and optimal relaxation weight?

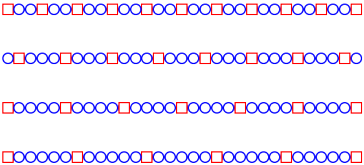
Training Dataset

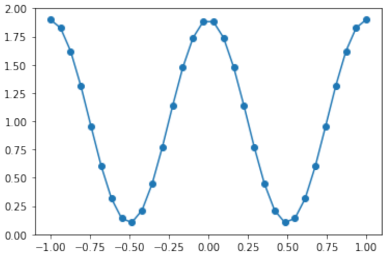
- For "traditional" machine learning we need a dataset.
- Idea: Run a *whole lot* of multigrid iterations.
- Run multigrid iterations and record convergence rate and relaxation weight for randomly generated C/F splittings and problem setups.



Dataset Generation

- Start from "reference" splittings – evenly spaced coarse points on grid
- Randomly perturb each reference in several trials, each point has a set probability of being flipped to opposite value
- Generate variable coefficients with a few random functions, i.e. cosine wave, random polynomial, noise

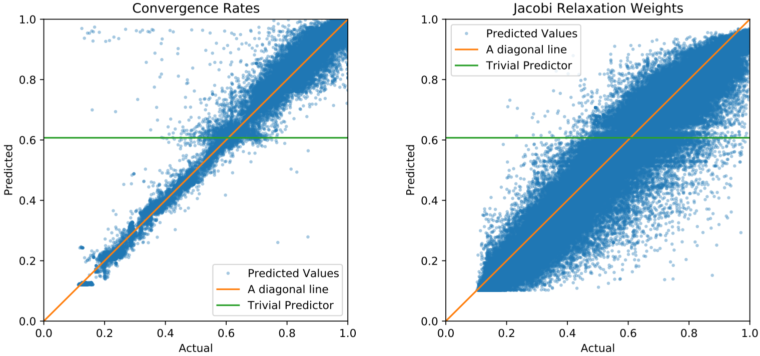




Multigrid, CNN

- Take the C/F splittings, run in multigrid solver to find convergence rate and relaxation weight that maximizes the former
  - Two level V-cycle solver, run for 50 iterations or until error sufficiently small
  - Two rounds Jacobi pre- and post-relaxation
  - Ideal 1D AMG interpolation operator: $\mathbf{P} = \begin{bmatrix} -\mathbf{A}_{FF}^{-1}\mathbf{A}_{FC} \\ \mathbf{I} \end{bmatrix}$
- Use the data to train a *1D convolutional network* that predicts convergence, Jacobi relaxation.
  - Look at neighboring values of nodes to predict features
  - Stack multiple CNN layers followed by fully-connected layer to force scalar output

CNN Performance



What we learned: Poisson is too easy!

Let's try learning a more difficult problem.

Convection-Diffusion Problem

Solve specific problem,

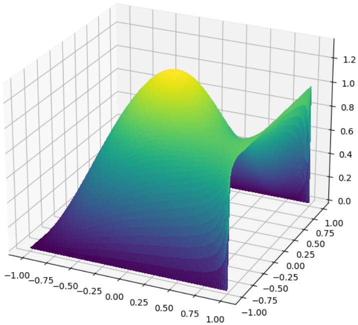
$$\mathbf{w} \cdot \nabla \mathbf{u} - k \nabla^2 \mathbf{u} = f$$

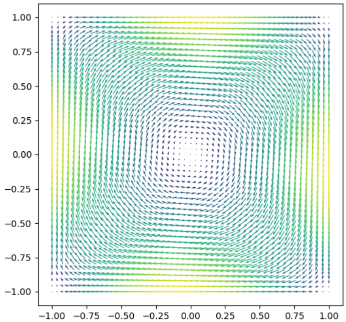
$$\Omega = [-1, 1]^2$$

$$k = 0.1$$

$$\mathbf{w} = \begin{bmatrix} 2y(1-x^2) & 2x(1-y^2) \end{bmatrix},$$

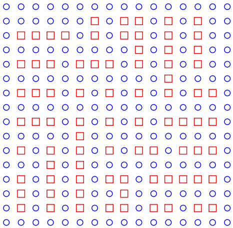
discretized as quad finite elements.

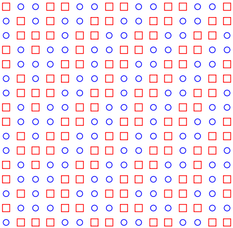




Dataset Generation, Convection-Diffusion

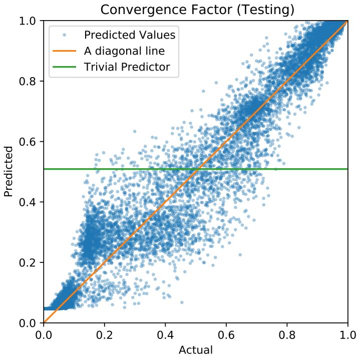
- Discretize on a 25×25 structured grid
- Start from "reference" splittings, all fine, all coarse, AMG output, etc
- Randomly perturb again in various trials
- Don't generate coefficient values for now
- Take output and run through 50 iteration multigrid solver to find convergence rate

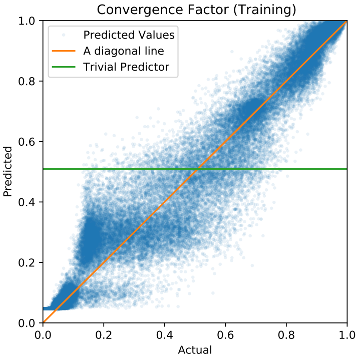




Convection-Diffusion Convolution

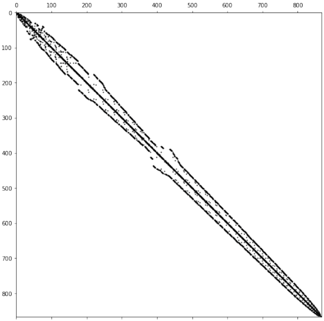
- 2D structured grid ⇒ train 2D convolutional network to predict convergence



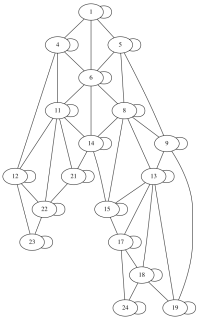


CNN to GNN

- Classical convolution techniques work okay on structured, grid-like inputs
- Very restrictive in terms of mesh data we can use for FEM solvers
- Take a look at some network architectures that allow for unstructured data: introduce *graph-nets*
  - Get FEM matrix, convert to graph and try to learn properties about the system

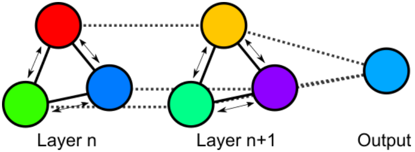


$\Longrightarrow$



Message-Passing Graph Convolutions

- Many graph convolution implementations, one such is the *Message-Passing Graph* layer
- In each layer, nodes learn optimal "messages" to pass via edges. Each node passes this message to other nodes in its neighborhood.
- Stacking multiple of these layers approximates traditional grid-based convolution.
- Run each set of nodal values through small fully-connected NN, take average for final convergence rate.

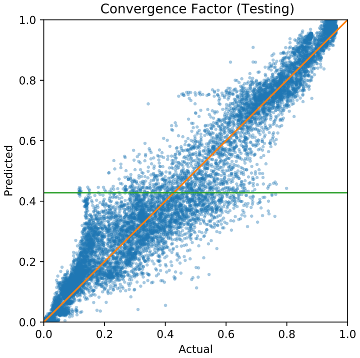


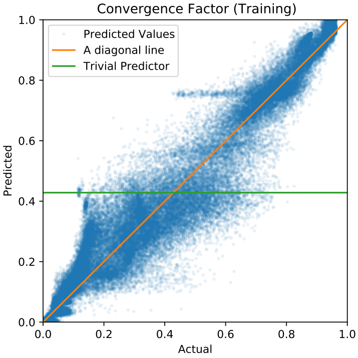
Message-Passing Dataset

- Decoupled the neural network from a fixed input size due to final aggregation step.
- Can have variable-sized input. Now generate and train on a variably-sized dataset of four mesh sizes:

$$\{15 \times 15 \quad 25 \times 25 \quad 35 \times 35 \quad 50 \times 50\}$$

Message-Passing Performance



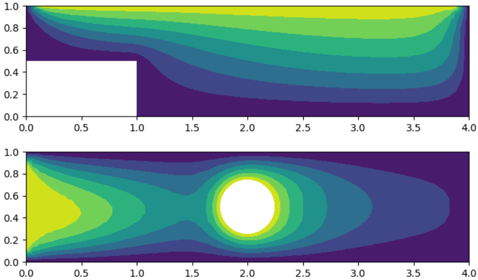


Conclusions

- These AMG features are indeed learnable with supervised methods.
- Predicting these values becomes more difficult on more complex problems, domains.
- What else can we learn?

Future Directions

- Try out some more interesting problems:



- Pick between different AMG methods with predictions.
- Use predictions in an optimization routine to find most convergent C/F splitting.